Exposing Business Objects as Web Service and Developing Web Dynpro Application

Applies to:

SAP Net Weaver CE 7.1.1 EHP1 (SP0, SP1, SP2 and SP3). For more information, visit the Web Dynpro Java homepage.

Summary

This article explains exposing Business Objects as Web Service in CAF and developing Web Dynpro Application using Adaptive Web Service Model.

Author: Sunil Kumar S S

Company: SAP Labs India Pvt. Ltd.

Created on: 04 May 2009

Author Bio

Sunil Kumar S S works as a Quality Engineer within the Web Dynpro Tools team. He has been associated with SAP Labs since July 2007. His areas of expertise include Web Dynpro Java, Data Dictionary, Web Services, CAF and NWDI.

Table of Contents

Prerequisite

- SAP Web Application Server Java 7.1 and authorization to configure service group.

- SAP NetWeaver Developer Studio 7.1

Introduction

CAF Business Objects

A business object is a semantic entity that represents the smallest data unit to take part in a business scenario.

You use a business object to define persistable data types and to generate database tables. You can specify relations between business objects, persistency and permissions, implement or map basic methods for reading, writing, updating, deleting or querying data.

A business object is a clearly identifiable unit.

Ex: For a car rental system, you can create the following business objects:

- Car – with attributes such as license plate number, car model, number of seats, maximum speed, mileage, color

- Customer – with attributes such as the first name, last name, credit card number, address, phone, fax, e-mail, date of registration

- Order – with attributes such as the date, invoice number, or price.

You should create associations with the other business objects to specify details about the customer and car.

Web Dynpro Adaptive Web Service Model

If you plan to reuse a Web Service which is either locally available or from a UDDI then the Adaptive Web Service Model is the interface you would use for your Java Web Dynpro application.

Modeling Business Objects

Creating Composite Application Development Component

Prerequisites

1. Launch the SAP NetWeaver Developer Studio.

2. Configure Application Server (AS) Java: Windows -> Preference -> SAP AS JAVA.

3. Choose Window -> Open Perspective -> Other.

4. Select Composite Application from the list.

5. Choose OK.

Procedure

1. Choose File -> New ->Project…

2. Select Development Infrastructure -> Development Component and choose Next.

3. Select Composite Application and click next

4. Select the software component for the new DC to be created in and Click next.

   The default software component is Local Development -> My Components.

5. Fill the required data as follows:

> *Vendor* **:** demo.sap.com
>
> *Name* **:** Enter the name of the DC.
>
> *Caption* **:** Enter a single line caption.
>
> *Language* **:** Choose the language. (*Default value:* American English)
>
> *Domain* **:** Choose the domain. (*Default value* **:** Basis)
>
> *Support Component* **:** BC

6. Choose Next, then choose Finish. The application you have created is added to the Composite Application Explorer.

Creating Business Objects

1. Click on the default package 'modeled' to create the business object with the secondary mouse button and choose New Business Object.

2. Enter a name for the business object.

3. Choose where you want to put your business object node, choose *Create a new structure*.

4. Choose Finish.

Implementing Permission Checks for Business Objects

1. Double Click on the Business Object created.

2. Open the Permissions tab and enable the *Permission checks enabled* indicator.

Exposing Business Objects as Web Service

To reuse the business logic stored in your application services or business objects, you can expose them as a Web service.

1. In the Composite Application Explorer, click the node of the business object you want to expose with secondary mouse button to choose *Expose Service as Web Service*.

2. In the Select service operation window, choose all the operations to expose.

3. Save, generate, build and deploy your application.

Find the WSDL URL of Exposed Web Service

1. Open Single Service Administration of SAP AS, http://<servername>:port/nwa/ssadmin

2. Provide user name and password to login.

3. In Service Definitions tab, enter the business object name in Find field and Click Go.

Note: There might be more than one Service definition with the same name listed in the table. Select the service definition which is relevant to your composite application DC.

4. Click on the WSDLs tab.

5. Copy the WSDL URL link and paste it in a notepad.

Create a Web Dynpro Application consuming web service exposed with CAF Business Objects

Creating a Web Dynpro Project

To manage local development objects of an application, you first need an appropriate project in the Developer Studio. Once you have set up the initial project framework, you can create the specific components of the Web application, define them, and implement the necessary source code. There is a wizard which generates an initial structure for your new Web Dynpro project. The Web Dynpro Explorer provides a logical view of the structure of a Web Dynpro project then. With this project view, you can access all Web Dynpro units. In addition, each Web Dynpro project provides access to local dictionaries, with the help of which you can create user-defined dictionary types. Additionally, you can add required projects resources, such as separate Java help classes, to the Web Dynpro project through the resources node.

Procedure

1. Select Development Infrastructure -> Development Component and choose Next.

2. Select Web Dynpro and Click next.

3. Select the software component for the new DC to be created in and Click next.

   The default software component is Local Development -> My Components.

4. In the window that appears, enter a name for the project and specify the settings for Project contents and

   Project language.

5. Finishing the wizard you are asked to open the perspective, based on the type of the project to be created

   is associated with the Web Dynpro perspective.

Creating the Application Unit

The project structure you have created does not yet include all the elements that enable you to define the specific functions of the whole Web Dynpro application. These additional elements are encapsulated in a Web Dynpro component that can contain any number of windows and views for visual representation and their corresponding controllers. Moreover, you also need an object that can be used to address and display the Web Dynpro component in the Web browser. This is why you create the logical unit application as an additional object. As an independent program unit it connects to an URL that can be accessed from the user via a window within the Web Dynpro component.

Procedure

1. Expand the Web Dynpro node in the structure of the Demo project.

2. To start the relevant wizard, open the context menu of the Applications node, and choose Create

   Application.

3. Enter a name for the application unit, such as unit *DemoApp*, and specify the package name (example:

   com.sap.demoapp) in which the generated Java classes will be stored. Choose Next.

4. Assign a Web Dynpro component to which the application is linked. Since a Web Dynpro component has

   not yet been created, the wizard creates a component at this point. Accept *Create a new component* and

   choose Next again.

5. In the next screen you specify some general values for the component. Enter the name *DemoComp* for

   your Web Dynpro component and specify the package name *com.sap.demoapp.comp*. Accept the other

suggested settings and choose Finish.

Result

The application wizard performs several generation routines. Once it has finished, it adds the new application object to the structure. Additionally, it inserts a substructure for the new Web Dynpro component. Furthermore the initial *DemoCompView* is automatically created for you; you find this view under the Views node in your Web Dynpro project.

Furthermore, a window unit with name <myproject>Window, in this example with name *DemoWindow*, has been generated; including the needed plug for application starting purposes.

Importing Adaptive Web Service Model

Using the Web Dynpro model import wizard, you address a Web Service definition which is available locally or from a UDDI for your Web Dynpro application.

Procedure

1. To start the relevant wizard, open the context menu of the Models node, and choose *Create Model.*

   In the model import wizard, choose the Adaptive Web Service Model option and Click next.

2. Enter a name for the model, such as unit *Demomodel*, and specify the package name (example:

   com.sap.demomodel).

3. From Available WDSL resources select Remote Location / File System, Click Next.

4. For the WSDL URL, enter the WSDL URL exposed as web service by business objects and click next.

5. Create New Service Group with default values and click next.

6. The importer displays the model classes that will be generated from the WSDL. Any model name clashes

   will be resolved. One can see the system renamed model classes by selecting the check box "Show

   Renamed Model Classes". Note this check box is only enabled if system renamed model classes exist.

7. Click Next, Import Log is shown.

8. Click Finish. Model is created and visible in the WD explorer.

Creating a User Interface to access data from web service

Apply Service Controller Template

Creates a controller that wraps a service. This service is either a Remote Function Call or a Web service. The wizard then creates the required contexts and an optional method for triggering the call.

Procedure

1. Add the created model to the list of Used models node under Component with the secondary mouse

   button, choose the created model and click Ok.

2. Select the View in Web dynpro explorer with the secondary mouse button, choose Template -> Apply

3. Select Service Controller and Click Next.

4. Select *Request_Create* Model Class and Click next.

5. Select all the nodes, attributes under *Request_Create* and Click next.

6. Enable Generate method for model execution and Click on Finish.

Apply Action Button Template

The wizard for the Action Button template allows you to easily create a button, an action, an event handler, and the code for calling a method and triggering an event.

Procedure

1. Select the View in Web dynpro explorer with the secondary mouse button, choose Template -> Apply

2. Choose the option *ActionButton*. Enter a name for the template or use the default.

3. Choose *Next*. In the next window, you use the default for action, and event. For the label, Enter *Create*.

4. Choose *Next* to proceed to the wizard window for defining the methods and plugs.

5. Select the check box Call Method and for controller drop down field select the View Controller.

   Corresponding method is also filled in Method drop down field.

6. Close the wizard by choosing *Finish* and save your changes.

Apply Form Template

You can use the form template wizard to create a form layout for a view, based on the view context. For designing and implementing this form layout, you can use controller context data or access the data of a declared Web Dynpro model directly.

Procedure

1. Select the View in Web dynpro explorer with the secondary mouse button, choose Template -> Apply

2. Choose the option Form. Enter a name for the layout template or use the default.

3. Choose Next. You see a window in which you can choose the context attributes you require for the form.

4. Select the nodes and attributes displayed in the image below.
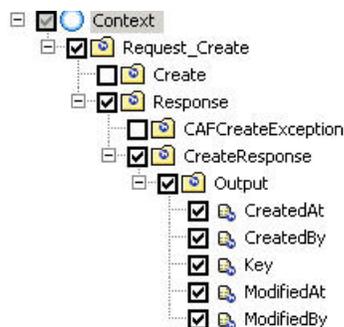
5. Close the wizard by choosing Finish and then choose Save All to save your changes.

UI Design

1. Double Click on the view designer.

2. Go to outline View and Select Transparent Container and go to properties View and change the property value layout to Grid Layout.

3. Select each label field in outline view and go to properties View and change the property value text as shown below.

Build and Deploy DCs

Build

1. On the web dynpro project with the secondary mouse button choose Development Component > Build.

2. Select all the DCs and Click OK. Make sure that Fore Build Check box is active.

Deploy

1. On the web dynpro project with the secondary mouse button choose Deploy.

2. Enter the J2ee Engine Username and password and click OK.

3. Click OK on Deploy wizard once deployment is successful.

Configure Service Group in SAP NetWeaver Administrator.

Create Communication Profile

1. Open NetWeaver Administrator page http:/<server name>:<port>/nwa

2. Click on SOA management.

3. Click on System Connections under Technical configuration tab.

4. Click on Communication Profiles tab. Click on New Button. Give some Profile Name (ex: demoprofile),

   Click Next, Next and Finish (maintain all default settings).

Create Provider System

1. Go to Provider Systems tab.

2. Click New,

   Select System type: Java

   Enter System Name: The SID of your system

   Host: Host address of your system .

3. Click Next, Next, select the Profile Name (ex: demoprofile) from dropdown menu.

4. Click Next, select Mode as Single Service and for WSDL URL, enter the WSDL URL exposed as web

   service by business objects. Click Finish.

Configuring Application communication on J2EE engine

1. Open SOA Management tab

2. Select Application and Scenario Communication -> Application Communication.

3. Select your deployed WD project

The Service Group should appear automatically in a table.

4. Select the Service Group & Click the Edit Button -> Click the "Assign Provider System" button -> Click on

   Go button and select the Provider System -> Click on save button.

Run Application

To deploy the logical Application unit which is also displayed in the Web Dynpro Explorer view, you select Deploy new Archive and Run. As you can see, you would also trigger the execution process for the WD application.

Output

```
Create
CreatedAt:
2009-04-30T6:37:35.867
CreatedBy:
J2EE_GUEST
Key:
a0bddcb2-3540-11de-9e0
ModifiedAt:
2009-04-30T6:37:35.867
ModifiedBy:
J2EE_GUEST
```

Related Content

User Interface Technology

Web Dynpro Java

Composite Application Framework

Copyright